17 marks programs

Write down program to reshape the 3 x 4 array into 3 dimensions of size 2,2,3 and 2,3,2 using numpy.

```
#New example - reshape

import numpy as np

arr =np.array([[1, 2, 3, 4],

[5, 2, 4, 2],

[1, 2, 0, 1]])

newarr = arr.reshape(2, 2, 3)

newarr1 = arr.reshape(2, 3, 2)

newarr2=arr.reshape(4,3)

print ("\nOriginal array:\n", arr)

print ("Reshaped array:\n", newarr)

print ("Reshaped array:\n", newarr1)

print ("Reshaped array:\n", newarr2)
```

#Faker Programs

1. **Write a program in python to print 10 fake names and countries in Hindi language using faker package.**

```
fake = Faker('hi_IN')
name=[]
country=[]
for _ in range(10):
   name.append(fake.name())
   country.append(fake.country())
print(name)
print(country)
```

2. **Write a program in python to create a profile and print a particular fake data.**

```
import faker
#signature
#simple_profile(sex: Optional[GenderType] = None) → Dict[str, Union[str, datetime.date, GenderType]]
faker_obj = faker.Faker()

gender = 'M'
male_profile = faker_obj.simple_profile(sex=gender)
print("Male profile generated is as follows:\n", male_profile)
```

```
gender = 'F'
female_profile = faker_obj.simple_profile(sex=gender)
print("Female profile generated is as follows:\n", female_profile)


gen_profile = faker_obj.simple_profile()
print("Combinations of profile is as follows:\n",gen_profile)
```

3. **Write a program in python to create a json of 100 students with name students.json that contains student name, address, location coordinates and student roll number using faker package.**

```
import json
n=[]
a=[]
loc=[]
for _ in range(100):
  n.append(fake.name())
  a.append(fake.address())
  loc.append(fake.location_on_land())
dt = {"name":n,"address":a,"location":loc}
json = json.dumps(dt)
#print("All json Data")
print(json)
```

#Numpy Prgrams

4. **Write a program in python using NumPy**
1. **to generate 10 random numbers from the normal distribution**
2. **to create a 3x3x3 array with random values**

```
import numpy as np
x = np.random.normal(size=10)
print(x)
x = np.random.random((3,3,3))
print(x)
```

5. **Write a program in python using NumPy to create a 5x5 array with random values and find the minimum and maximum values**

```
import numpy as np
x = np.random.random((5,5))
print("Original Array:")
print(x)
xmin, xmax = x.min(), x.max()
print("Minimum and Maximum Values:")
print(xmin, xmax)
```

6. **Write a program in python using NumPy to create a random vector of size 10 and sort it**

```
import numpy as np
x = np.random.random(10)
```

```
print("Original array:")
print(x)
x.sort()
print("Sorted array:")
print(x)
```

7. Write a NumPy program to check two random arrays are equal or not.

```
import numpy as np
x = np.random.randint(0,2,6)
print("First array:")
print(x)
y = np.random.randint(0,2,6)
print("Second array:")
print(y)
print("Test above two arrays are equal or not!")
array_equal = np.allclose(x, y)
print(array_equal)
```

8. Write a program in python using NumPy to find the most frequent value in an array

```
import numpy as np
x = np.random.randint(0, 10, 40)
print("Original array:")
print(x)
print("Most frequent value in the above array:")
print(np.bincount(x).argmax())
```

9. Write a program in python using NumPy to get the 3rd largest values of an array.

```
import numpy as np

arr = np.array([2, 0, 1, 5, 4, 1, 9])
print("Given array:", arr)
sorted_index_array = np.argsort(arr)
sorted_array = arr[sorted_index_array]
print("Sorted array:", sorted_array)

# we want 3rd largest value
n = 3

rslt = sorted_array[-n : ]

print("{} largest value:".format(n), rslt[0])
```

#Output

```
Given array: [2 0 1 5 4 1 9]
Sorted array: [0 1 1 2 4 5 9]
   1  largest value: 9
```

10.Write a program in python using Matplotlib to display a bar chart of the popularity of programming Languages.

```
import matplotlib.pyplot as pyplot

labels = ('Python', 'Java', 'JavaScript', 'C#', 'PHP', 'C,C++', 'R')
sizes = [29.9, 19.1, 8.2, 7.3, 6.2, 5.9, 3.7]

pyplot.bar(labels, sizes, color="#6c3376")

pyplot.ylabel('Usage in %')
pyplot.xlabel('Programming Languages')

pyplot.show()
```

11.Write a program in python using Matplotlib to display a horizontal bar chart of the popularity of programming Languages.

```
import matplotlib.pyplot as pyplot

labels = ('Python', 'Java', 'JavaScript', 'C#', 'PHP', 'C,C++', 'R')
sizes = [29.9, 19.1, 8.2, 7.3, 6.2, 5.9, 3.7]

pyplot.barh(labels,sizes, color="#6c3376")

pyplot.ylabel('Usage in %')
pyplot.xlabel('Programming Languages')

pyplot.show()
```

12. Write a program in python using Matplotlib to display a bar chart of the popularity of programming Languages. Use different color for each bar

```
import matplotlib.pyplot as plt

x = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']

popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]

plt.bar(x, popularity, color=['red', 'black', 'green', 'blue', 'yellow',
'cyan'])

plt.xlabel("Languages")

plt.ylabel("Popularity")

plt.title("PopularitY of Programming Language\n" + "Worldwide, Oct 2017
compared to a year ago")

plt.show()
```

13. Write a program to take the screenshot and print the characteristics of image – width and height and rotate the image by 90 degree.

```
from PIL import Image,ImageGrab
import PIL
image1 = ImageGrab.grab(bbox=None)
image1.show()
width, height = image1.size
print(width, height)
im1 = image1.rotate(90)
im1.show()
```

14. Write a program to extract metadata of given pdf file.

```
import pikepdf
pdf_filename = '/content/gdrive/My Drive/digital forensics/Practical-Cryptography.pdf'
pdf = pikepdf.Pdf.open(pdf_filename)
docinfo = pdf.docinfo
for key, value in docinfo.items():
    print(key, ":", value)
```

15.Write a program to extract the information like “from , date and subject” from email file

```
import email, email.policy
with open('C:/Users/Madhavi/No Food at Desk Policy.eml', 'rb') as f:
    msg = email.message_from_bytes(f.read(), policy=email.policy.SMTPUTF8)

    print("Message is from-: ",msg['from'])
    print("Message Date-: ",msg.get('date'))
    print("Message Subject-:",msg['Subject'])
```

16 . Write a program to do encryption using ROT12 algorithm

```
_
rot13trans = str.maketrans('NOPQRSTUVWXYZABCDEFGHIJKLMnopqrstuvwxyzabcdefghijklm',
  'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz')


# Function to translate plain text
def rot13(text):
  return text.translate(rot13trans)
def main():
  txt = "VaqveN Pbyyrtr"
  print (rot13(txt))

if __name__ == "__main__":
```

main()

17. Write a program to get username and sid to access the $R and $I file of recycle bin(This program you are supposed to run on Windows)

First Program – **recy1.py**

To get the username and sid to access the $R and $I file of Recycle Bin

Which command is used to the username and sid to access the $R and $I file of Recycle Bin ? - 2 marks
Ans - wmic useraccount get name,sid

```
import subprocess
username = "Administrator"
user_sid = "S-1-5-21-2668277920-2088842545-1183312488-500"
wmic_query = "wmic useraccount get name,sid"
user_sid = subprocess.check_output(wmic_query, shell=True)
print(type(user_sid))
str1 = user_sid.decode('UTF-8')
print(type(str1))
print(str1)
```

18. Write a program to take the screenshot , crop the image, change the size and display and save all newly created images.

```
from PIL import Image,ImageGrab
image2 = ImageGrab.grab(bbox=None)
image2.show()
width, height = image2.size
print(width, height)
left = 4
top = height / 5
right = 154
bottom = 3 * height / 5
 im1 = image2.crop((left, top, right, bottom))
im1.show()
newsize = (300, 300)
im1 = im1.resize(newsize)
 im1.show()
```

19. Write a program to extract metadata from mp4 file

```
from tinytag import TinyTag
def handle_mp4file(mp4_file):
        video = TinyTag.get(mp4_file)
        print("Title:",video.title)
        print("composer:",video.composer)
```

```
        print("bitrate:",video.bitrate)
        print("size:",video.filesize)
        print("Duration:",video.duration)
        print("Genre:",video.genre)

file_path = '/content/gdrive/My Drive/digital forensics/Free_Test_Data_15MB_MP4.mp4'
handle_mp4file(file_path)
```

20. **Write a program to demonstrate to save two numbers in mat format using scipy**

```
import scipy.io as syio
# Save the mat file
n = 1706256
n1= 23456
syio.savemat('num.mat', {'num': n,'num1':n1})
# Load the mat File
matlab_file_contents = syio.loadmat('num.mat')
print(matlab_file_contents['num1'])
print(matlab_file_contents)
# printing the contents of mat file.
matlab_file_contents = syio.whosmat('num.mat')
print(matlab_file_contents)
```

Large Programs – 28 marks programs

1. **Write a program in python using Matplotlib to display a bar chart of the popularity of programming Languages. Attach a text label above each bar displaying its popularity (float value)**

```
import matplotlib.pyplot as plt

x = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']

popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]

fig = plt.figure()

ax = fig.add_axes([0.1, 0.1, 1.0, 1.6])

rects1 = ax.bar(x, popularity, color=['red', 'black', 'green', 'blue',
'yellow', 'cyan'])

plt.xlabel("Languages")

plt.ylabel("Popularity")

plt.title("PopularitY of Programming Language\n" + "Worldwide, Oct 2017
compared to a year ago")

def autolabel(rects):

  for rect in rects:

    height = rect.get_height()
```

```
    ax.text(rect.get_x() + rect.get_width()/2., 1.05*height,'%2.3f' %
float(height),ha='center', va='bottom')

autolabel(rects1)

plt.show()
```

2. Write a program to create a report in csv using list and dictionary object

```
import csv

fields = ['Name', 'Branch', 'Year', 'CGPA']
rows = [ ['Mira', 'Civil', '2', '9.0'],
        ['Radha', 'Mechanical', '2', '9.1'],
        ['Krushna', 'IT', '2', '9.3'],
        ['Rukhmini', 'Computer', '1', '9.5'],
        ['Neil', 'Electrical', '3', '7.8'],
        ['Parth', 'ENTC', '2', '9.1']]
filename = "uni_records.csv"
with open(filename, 'w') as csvfile:
   csvwriter = csv.writer(csvfile)
   csvwriter.writerow(fields)
   csvwriter.writerows(rows)


mydict =[{'branch': 'COE', 'cgpa': '9.0', 'name': 'Nikhil', 'year': '2'},
        {'branch': 'COE', 'cgpa': '9.1', 'name': 'Sanchit', 'year': '2'},
        {'branch': 'IT', 'cgpa': '9.3', 'name': 'Aditya', 'year': '2'},
        {'branch': 'SE', 'cgpa': '9.5', 'name': 'Sagar', 'year': '1'},
        {'branch': 'MCE', 'cgpa': '7.8', 'name': 'Prateek', 'year': '3'},
        {'branch': 'EP', 'cgpa': '9.1', 'name': 'Sahil', 'year': '2'}]

fields = ['name', 'branch', 'year', 'cgpa']
filename = "university_records1.csv"
with open(filename, 'w') as csvfile:
   writer = csv.DictWriter(csvfile, fieldnames = fields)
   writer.writeheader()
   writer.writerows(mydict)
```

3. Write a program to create a excel sheet using dictionary object.

```
import xlsxwriter

workbook = xlsxwriter.Workbook('write_dict.xlsx')
worksheet = workbook.add_worksheet()

my_dict = {'Radha': [10, 11, 12],
         'Mira': [20, 21, 22],
         'Krishna': [30, 31, 32]}
```

```
col_num = 0
for key, value in my_dict.items():
   worksheet.write(0, col_num, key)
   worksheet.write_column(1, col_num, value)
   col_num += 1

workbook.close()
```

4. Program to create a plist file using dictionary object and display its contents (write 2 different programs for it)

```
import datetime
import plistlib

pl = dict(
   aString = "Doodah",
   aList = ["A", "B", 12, 32.1, [1, 2, 3]],
   aFloat = 0.1,
   anInt = 728,
   aDict = dict(
      anotherString = "<hello & hi there!>",
      aThirdString = "M\xe4ssig, Ma\xdf",
      aTrueValue = True,
      aFalseValue = False,
   ),
   someData = b"<binary gunk>",
   someMoreData = b"<lots of binary gunk>" * 10,
   aDate = datetime.datetime.now()
)
filename = "a.plist"
with open(filename,'wb')as fp:
plistlib.dump(pl,fp)
```

Program to read the contents of plist file

```
import plistlib
filename = "a.plist"
with open(filename,'rb')as fp:
   p1 = plistlib.load(fp)

print("All content of the file a.plist :-")
print(p1)
print("\n\nContent of 'aDict' key :-")
```

```
print(p1['aDict'])
```

5. Write a program to read and display the metadata of provided mp3 file.

```
import mutagen
def handle_id3(id3_file):
  id3_frames = {'TIT2': 'Title', 'TPE1': 'Artist', 'TALB': 'Album','TXXX':
    'Custom', 'TCON': 'Content Type', 'TDRL': 'Date released','COMM': 'Comments',
      'TDRC': 'Recording Date'}
  print("{:15} | {:15} | {:38} | {}".format("Frame", "Description","Text","Value"))
  print("-" * 85)

  for frames in id3_file.tags.values():
    frame_name = id3_frames.get(frames.FrameID, frames.FrameID)
    desc = getattr(frames, 'desc', "N/A")
    text = getattr(frames, 'text', ["N/A"])[0]
    value = getattr(frames, 'value', "N/A")

    if "date" in frame_name.lower():
      text = str(text)
    print("{:15} | {:15} | {:38} | {}".format(
      frame_name, desc, text, value))

file_path = '/content/gdrive/My Drive/digital forensics/01 - Aaj Dil Shayaraana
[FreshMaza.Info].mp3'
av_file = mutagen.File(file_path)
handle_id3(av_file)
```

6. Write a program to extract metadata from image file

```
from PIL import Image
from PIL.ExifTags import TAGS
import sys

fpath='/content/gdrive/My Drive/digital forensics/manya1.jpg'
img_file = Image.open(fpath)
exif_data = img_file._getexif()
#print(exif_data)
if exif_data is None:
  print("No EXIF data found")
  sys.exit()
print('\n Information Regarding Image \n')
print('Tag Name\tName\tValue')
for name, value in exif_data.items():
  gps_tag = TAGS.get(name, name)
```

```
print(gps_tag,'\t',name,'\t',value)
if gps_tag is not 'GPSInfo':
  continue
print(type(value))
print(name,value)
print("\n Now continue with other values\n\n")
```

7. Create two tables – student and department using sqlite and write a program to create csv file of student table using sqlite backup file.

Command to enter in sqlite -

sqlite database1.db

Command to create tables:-

create table student(std_id int primary key, std_name text, address text, ph_no char(10))

create table department(dept_id int primary key, dept_name text, location text)

Command to insert the values in student table:-

insert into student
values(1,'Sarita','Dehuroad','6767291987');
insert into student
values(1,'Prajwal','Thergoan','7797291987');
insert into student
values(1,'Ninad','Sangavi','987291987');

Command to take backup

.backup std.db

Program to create a csv file.

```
import csv
import sqlite3

conn = sqlite3.connect('std.db')
cursor = conn.cursor()
cursor.execute("select * from Student;")
with open("out.csv", 'w',newline='') as csv_file:
   csv_writer = csv.writer(csv_file)
   csv_writer.writerow([i[0] for i in cursor.description])
   csv_writer.writerows(cursor)
conn.close()
```

8. Create two tables – student and department using sqlite and write a program to create csv file of all table information(metadata of database) from sqlite backup file.

Command to enter in sqlite -

sqlite database1.db

Command to create tables:-

create table student(std_id int primary key, std_name text, address text, ph_no char(10))

create table department(dept_id int primary key, dept_name text, location text)

Command to insert the values in student table:-

insert into student
values(1,'Sarita','Dehuroad','6767291987');
insert into student
values(1,'Prajwal','Thergoan','7797291987');

insert into student
values(1,’Ninad’,’Sangavi’,’9877291987’);

Command to take backup

.backup std.db

Program to create a csv file.

```
import csv
import sqlite3

conn = sqlite3.connect('std.db')
cursor = conn.cursor()
cursor.execute("select * from sqlite_master where type='table';")
with open("alltables.csv", 'w',newline='') as csv_file:
   csv_writer = csv.writer(csv_file)
   csv_writer.writerow([i[0] for i in cursor.description])
   csv_writer.writerows(cursor)
conn.close()
```

9. Write a program to change the MAC address of the machine

```
import random
import os
import subprocess

def get_rand():
   return random.choice("abcdef0123456789")

def new_mac():
   new_ = ""
   for i in range(0,5):
      new_ += get_rand() + get_rand() + ":"
   new_ += get_rand() + get_rand()
   return new_

print(os.system("ifconfig eth0 | grep ether | grep -oE [0-9abcdef:]{17}"))
subprocess.call(["sudo","ifconfig","eth0","down"])
new_m = new_mac()
subprocess.call(["sudo","ifconfig","eth0","hw","ether","%s"%new_m])
subprocess.call(["sudo","ifconfig","eth0","up"])
print(os.system("ifconfig eth0 | grep ether | grep -oE [0-9abcdef:]{17}"))
```

10. **Write a program to connect to Google using socket**

```
# Write a program to connect to Google using socket
import socket # for socket
import sys

try:
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    print("Socket successfully created")
except socket.error as err:
    print("Socket creation failed with error %s" %(err))

#default port for socket
port = 80

try:
    host_ip = socket.gethostbyname('www.google.com')
except socket.gaierror:
    print("There was an error resolving the host")
    sys.exit()

#connecting to the server
s.connect((host_ip, port))
print("the socket has successfully connected to google")
```

11. **Write a program using command line argument to display location, region, city and country of website**

```
import sys
import requests
import socket
import json

if len(sys.argv) < 2:
    print("Usage: " + sys.argv[0] + "<url>")
    sys.exit(1)

req = requests.get("https://"+sys.argv[1])
print("\n"+str(req.headers))

gethostby_ = socket.gethostbyname(sys.argv[1])
print("\nThe IP address of "+sys.argv[1]+" is: "+gethostby_ + "\n")

req_two = requests.get("https://ipinfo.io/"+gethostby_+"/json")
resp_ = json.loads(req_two.text)

print("Location: "+resp_["loc"])
print("Region: "+resp_["region"])
print("City: "+resp_["city"])
print("Country: "+resp_["country"])
```

12. Write a packet sniffer program to display source address, destination address and protocol used

```
import socket
import struct
import sys
def get_mac_addr(bytes_addr):
    bytes_str = map('[:02x]'.format, bytes_addr)
    return ':'.join(bytes_str).upper()

def ethernet_head(raw_data):
    dest, src, prototype = struct.unpack('! 6s 6s H',raw_data[:14])
    dest_mac = get_mac_addr(dest)
    src_mac = get_mac_addr(src)
    proto = socket.htons(prototype)
    data = raw_data[14:]
    return dest_mac, src_mac, proto, data
def main():
    s = socket.socket(socket.AF_PACKET, socket.SOCK_RAW, socket.ntohs(3))
    while True:
        raw_data, addr = s.recvfrom(65535)
        eth = ethernet_head(raw_data)
        print('\n Ethernet FRame:')
        print('Destination: {}, Source: {}, Protocol: {}'.format(eth[0], eth[1], eth[2]))
```

13. write down the program to explain the encryption of Caesar Cipher

```
text = "INDIRACOLLEGExyz"
s = 4
result =""
for i in range(len(text)):
    char = text[i]
    # Encrypt uppercase characters in plain text
    if (char.isupper()):
      r = ord(char)+4
      if r >= 91:
        r = 65 + (r-91 )
      result += chr(r)
    # Encrypt lowercase characters in plain text
    else:
      r = ord(char)+4
      if r >= 123:
        r = 97 + (r-123)
        result += chr(r)
print(result)
```

14. Write down the program for Hacking of Caesar Cipher

```
message = 'MRHMVEGSPPIKI' #encrypted message
LETTERS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

print("encrypted message : ",message)
print("\n\n")

for key in range(len(LETTERS)):
  translated = ''
  for symbol in message:
    if symbol in LETTERS:
      num = LETTERS.find(symbol)
      num = num - key
      if num < 0:
        num = num + len(LETTERS)
      translated = translated + LETTERS[num]
    else:
      translated = translated + symbol
  print('Hacking key #%s: %s' % (key,translated))
```

15. Write down the program to do the encryption using Transposition cipher.

```
 def main():
  myMessage = 'Transposition Cipher'
  myKey = 4
  ciphertext = encryptMessage(myKey, myMessage)

  print("Cipher Text is")
  print(ciphertext)


def encryptMessage(key, message):
        ciphertext = [''] * key
        for col in range(key):
                position = col
                while position < len(message):
                        ciphertext[col]+=message[position]
                        print(col)
                        position += key
                print(ciphertext)
        return ''.join(ciphertext)

if __name__ == '__main__':
  main()
```

16. **Write down the program to do the encryption using Transposition cipher.**

```
import binascii
input_str = input("Enter the cipher text or plain text:")
key = input("Enter the key for encryption or decryption:")
no_of_itr = len(input_str)
output_str = ""

for i in range(no_of_itr):
   current = input_str[i]
   current_key = key[i%len(key)]
   print("curreent key = ", current_key)
   print( "Ascii value - letter", ord(current), " key", ord(current_key))
   output_str += chr(ord(current) ^ ord(current_key))

print ("Here's the output: ", output_str)
```

17. **Write program(s) to establish a connection between server & client using socket (write 2 programs – server side and client side)**

#Server side program

```
import socket
def Main():
   host = socket.gethostname()
   port = 12345
   serversocket = socket.socket()
   serversocket.bind((host,port))
   serversocket.listen(1)
   print('socket is listening')

   while True:
      conn,addr = serversocket.accept()
      print("Got connection from %s" % str(addr))
      msg = 'Connecting Established'+ "\r\n"
      conn.send(msg.encode('ascii'))
      conn.close()
if __name__ == '__main__':
   Main()
```

//Client side program

```
import sockets = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
#AF_INET - it means ipv4, as opposed to ipv6 or domain notation, like 'daring.cwi.nl# SOCK_STREAM - TCP socket

host = socket.gethostname()
port = 12345
s.connect((host, port))
msg = s.recv(1024)
s.close()
print (msg.decode('ascii'))
```

18. Write a program to get the path of files present in recycle bin for given user name and sid.

```
import subprocess
import os
username = "Madhavi"
wmic_query = "wmic useraccount where name=\"" + username + "\" get sid"
user_sid = subprocess.check_output(wmic_query, shell=True)
print ("User id before strip %s." % user_sid)
user_sid = user_sid[4:].strip()
if user_sid == "":
        print ("Unable to retrieve user SID.")
        user_sid = input("Enter SID manually, or press return: ")
        if user_sid == "":
          raise Exception
uid = user_sid.decode()
print ("User SID is %s." % user_sid)
print ("Decoded User SID is %s." % uid)
print ("")
windows_drive="C"
recycled_directory =  "C:\$Recycle.Bin\\" + uid + "\\"
print ("Recycle Bin directory is %s." % str(recycled_directory))
recycled_files = os.listdir(recycled_directory)
for deleted_file in recycled_files:
        if deleted_file[1] == "I":
                full_path = recycled_directory + deleted_file;
                deleted_file_content = open(full_path, "r");
                deleted_file_path = deleted_file_content.read();
                deleted_file_content.close();
                deleted_file_path = deleted_file_path[28:];
                print(deleted_file_path)
```

19. Write a program to get the contents of log based artifacts of windows (run on windows)

```
import win32evtlog # requires pywin32 pre-installed
server = 'localhost' # name of the target computer to get event logs
logtype ='System' # 'Application' # 'Security' System
hand = win32evtlog.OpenEventLog(server,logtype)
flags = win32evtlog.EVENTLOG_BACKWARDS_READ|win32evtlog.EVENTLOG_SEQUENTIAL_READ
total = win32evtlog.GetNumberOfEventLogRecords(hand)
i=0
while True:
     events = win32evtlog.ReadEventLog(hand, flags,0)
     if (i>10):
         break
     i=i+1
     if events:
          for event in events:
                print ('Event Category:', event.EventCategory)
                print ('Time Generated:', event.TimeGenerated)
                print ('Source Name:', event.SourceName)
                print ('Event ID:', event.EventID)
                print ('Event Type:', event.EventType)
                print("Value of i=",i)
                data = event.StringInserts
                j=0
                if data:
                    print ('Event Data:')
                    for msg in data:
                        if(j==5):
                            break
                        j=j+1
                        print (msg,'\n')
```

20. Write a program for ping sweeping ICMP Protocol

```
import subprocess
import platform
from datetime import datetime

def ping_sweep(network, start, end):
    # Determine the operating system and set the ping command
    oper = platform.system()
    if oper == "Windows":
        ping_cmd = ["ping", "-n", "1", "-w", "1000"]
    else:
        ping_cmd = ["ping", "-c", "1", "-W", "1"]

    # Base network address
    base_network = '.'.join(network.split('.')[:3]) + '.'

    # Record the start time
    start_time = datetime.now()
    print("Scanning in Progress:")
```

```
    # Iterate over the range of IP addresses
    for ip in range(start, end + 1):
        addr = base_network + str(ip)
        cmd = ping_cmd + [addr]
        print(f"Pinging: {addr} with command: {' '.join(cmd)}")

        try:
            # Execute the ping command
            response = subprocess.run(cmd, stdout=subprocess.PIPE, stderr=subprocess.PIPE, text=True)
            # Check if 'TTL' is in the response
            if "TTL" in response.stdout or "ttl" in response.stdout:
                print(f"{addr} --> Live")
        except Exception as e:
            print(f"Error pinging {addr}: {e}")
```

```
# Get user inputs
network = input("Enter the Network Address (e.g., 192.168.1.0): ")
start = int(input("Enter the Starting Number: "))
end = int(input("Enter the Last Number: "))

# Perform the ping sweep
ping_sweep(network, start, end)
```